AlwaysUp Command-Line Tools Manual

Version 9.1

0. Version History 2
1. Introduction 3
2. Key Features & Benefits 4
3. System Requirements 5
4. InstallService 6
   4.1. Command Line Syntax 6
   4.2. Command Line Options 7
      4.2.1. General 7
      4.2.2. Logon 9
      4.2.3. Monitoring 10
      4.2.4. Customized “Sanity Checks” 12
      4.2.5. Notification (via E-Mail) 13
      4.2.6. Startup 15
   4.3. Return Codes 15
   4.4. Examples 16
   4.5. Using the AlwaysUp GUI to Create the Command Line 16
   4.6. Using the AlwaysUp GUI to Create an XML file 18
5. UninstallService 19
   5.1. Command Line Syntax 19
   5.2. Return Codes 19
   5.3. Examples 19
   5.4. Using SC 19
6. AlwaysUpService 21
   6.1. Export to XML 21
   6.2. Start or Restart your Application in the Current Session 21
7. Starting and Stopping AlwaysUp Services with the NET Command 23
   7.1. To Start a Service 23
   7.2. To Start a Service in a Specific Session 23
   7.3. To Start a Service in a Specific User’s Session 24
   7.4. To Stop a Service 24
8. Files to Distribute with your Applications 26
9. Integrating AlwaysUp CLT into your Application’s Installer 27
10. Licensing & Registration 28
11. Troubleshooting and Reporting Problems 29
12. Appendix I: The Sanity Check Utility 30
   12.1. Special Command Line Variables 31
   12.2. Checking a Web Server with http-ping 32
   12.3. Other Sanity Check Utilities 32
13. Appendix II: The InstallService XML 33
14. Appendix III: Remote Desktop and Terminal Services Sessions 35
15. Appendix IV: AlwaysUp Web Service 36
16. Appendix V: AlwaysUp Event Log Messages 37

Copyright © 2001-2015, Core Technologies Consulting, LLC

0. Version History

Version	*Date*	*Updates*
9.0	2014-11-02	Options to stop running instances, prevent sleep, minimize logging on delayed restarts. Removed the option to prevent close on logoff.
8.8	2014-06-15	Bug fixes.
8.7	2014-05-07	Stop/restart when the computer resumes.
8.6	2013-07-02	Registry entries to control event log writing.
8.5	2013-02-22	Option to pause on startup only when booting; Additional command line replacements.
8.0	2012-07-28	Added support for launcher applications.
7.9	2012-01-07	Added support for sending email with TLS or STARTTLS encryption (for Office 365); Added the ability to run a command when the application stops.
7.9	2011-06-25	Added options to send email when the service starts.
7.8	2011-02-10	Added options to delay the sanity check
7.5	2010-08-02	Added support for disabled services; added a section on Installers
7.0	2010-02-21	Intelligent restarting/rebooting; wait time when stopping; support the "Automatic (Delayed Start)" option; support for suspending the computer while waiting to restart the application
6.8	2008-12-01	Added support for starting/restarting in specific sessions, sanity check reboots, capping restart attempts.
6.7	2008-06-01	Plain text mail format support.
6.5	2007-11-01	Startup pause.
6.4	2007-10-04	Start “in session”; Ensure network is up option; Stop command; Capture output; Automation.
6.2	2007-08-19	New features: The main window can be adjusted on startup (minimized/maximized/normally; on top).
6.0	2007-03-12	Support Windows Vista; Site & OEM Branding.
5.7	2006-10-29	New features: Hide UI, restart-delay, etc.
5.5	2006-08-25	New features: XML support, restore network drives, prevent close at logoff.
5.0	2006-05-01	Initial version (from previous plain-text format)

Copyright © 2001-2015, Core Technologies Consulting, LLC

1. Introduction

The AlwaysUp command-line tools (CLT) are designed to run and manage any Win32 executable, batch file, or shortcut as a Windows service. Developers can easily integrate and distribute them as a part of their own applications.

Find out more about AlwaysUp CLT at:

http://www.coretechnologies.com/products/AlwaysUp/AlwaysUpCLT/

Copyright © 2001-2015, Core Technologies Consulting, LLC

2. Key Features & Benefits

- Runs your application (32/64-bit executable, batch file or shortcut) as a Windows Service
- Monitors your application and restarts it whenever it fails
- Able to manage virtually any Win32 application with minimal configuration
- Able to detect and restart applications that hang, hog the CPU or consume too much memory.
- Able to start your application as the system boots, to run when no user is logged in and to run despite logon/logoffs -- all to guarantee uptime without manual user intervention
- Able to restart your application (or reboot the computer) at a scheduled time
- Able to boost your application's priority to ensure that it gets preferential treatment on the host computer
- e-mails you with details of crashes, restarts and other problems
- e-mails you daily or weekly reports on your application's performance
- Supports the integration of your own custom "sanity check" utilities, executed regularly to test if your application is functioning normally or not
- Automatically dismisses common "Application error" dialog boxes that prevent crashed applications from fully exiting
- Automatically dismisses Debug dialog boxes and logs the dialog box text for subsequent review by developers -- thus facilitating the deployment of applications in "Debug mode"
- Leverages the Windows Services API to achieve robust, low-level application control in a secure environment
- Reports all activities to the Windows Event Log
- Supports automation, to automatically fill in forms, dismiss dialogs, etc.
- Captures output from console applications.
- Very efficient; demands minimal CPU & memory resources
- No programming required!
- Easily integrates into installers

There are three programs in the suite, namely **InstallService**, **UninstallService**, **AlwaysUpService**. Section 4 describes these programs in detail.

Copyright © 2001-2015, Core Technologies Consulting, LLC

3. System Requirements

- A Pentium-compatible PC
- Windows 8.1 / 8 / 7 / Vista / XP or Windows Server 2012 / 2008 / 2003 (x86 and x64 versions).
- 5 MB free hard drive space for installation files.

The AlwaysUp components that manage and monitor your application are designed to be extremely frugal with machine resources. They almost always consume less than 1% of the CPU, less than 8 MB of RAM, and don't fall victim to the "Memory Growth" characteristic of many applications today.

Copyright © 2001-2015, Core Technologies Consulting, LLC

4. InstallService

InstallService is used to install an application to run as a Windows Service.

4.1. Command Line Syntax

The command line syntax for InstallService is:

InstallService.exe <Service name> <App> [Options]

or, to install using an XML file:

InstallService.exe -xml <XML file name>

where:

<Service name> is the name of the service. The maximum length is 200 characters. Forward-slash (/) and back-slash (\) can not be used.

<App> is the full command line for running the application to be monitored. If the command line contains spaces, care must be taken to ensure that it is interpreted correctly. Be sure to enclose each component that contains spaces in “backslash quotes” (\”), and then quote the whole command line. For example:

C:\Program Files\Server\Server.exe -config C:\My Files\config.ini

should be expressed as:

“\”C:\Program Files\Server\Server.exe\” -config \”C:\My Files\config.ini\””

<XML file name> is the full path to an existing XML file describing the service. Additional details about the XML format is available in Appendix II.

[Options] are one or more options described in the next section.

Note: Be sure to use quotes around parameters containing spaces.

Copyright © 2001-2015, Core Technologies Consulting, LLC

4.2. Command Line Options

To facilitate feature-driven access, we present the command line options grouped by functionality.

4.2.1. General

-w	• Specifies the application's working directory. Be sure to enclose paths that contain a space in quotes (“).
-n	• Start the service/application “Manually” from the Service Control Manager (or the GUI-based AlwaysUp). • If neither -n, -nd, or -nx is specified, the behavior will default to “Automatic”, and the service/application will be started when Windows boots.
-nd	• Configure the service/application to start “Automatically (Delayed Start)”, 1-2 minutes after Windows boots. This option is only meaningful on Windows Vista and later.
-nx	• Disable the service. It will not be able to start/run.
-dp <mask>	• On multi-processor systems, restrict the application to run on one or more specific CPUs. • <mask> is a bit vector representing the chosen CPUs. Each bit represents a CPU, with: 1 = CPU #1 2 = CPU #2 4 = CPU #3 8 = CPU #4 (etc.) • For example, to specify CPU #1 and CPU #4, the <mask> value is 1 + 8 = 9. To select CPU #1, #2 and #3, use 1 + 2 + 4 = 7.
-f <priority> <reduce>	• <priority> specifies the application's priority: Idle = 1 BelowNormal = 2 Normal = 3 Above High = 5 Real-time = 6 • <reduce> indicates if the application's priority should be reduced to Normal when a user logs on to the computer: No = 0 Yes = 1 • If not specified, the application is run at Normal priority.

Copyright © 2001-2015, Core Technologies Consulting, LLC

-mm	• Show the application's main window minimized when it starts.
-mx	• Show the application's main window maximized when it starts.
-mm	• Show the application's main window normally when it starts.
-kt	• Keep the application's main window on top of others.
-st <cmd line>	• Use the given command to stop the application. • <cmd line> is the full command line for the program to run, including any arguments. Please use quotes around each element if the command line contains spaces. • AlwaysUp will wait up to 30 seconds for the command to complete before terminating the application. • See section 11.1 for special command-line values.
-sw <seconds>	• Wait up to <seconds> seconds for the application to stop gracefully before AlwaysUp forcibly terminates it.
-sc < cmd line>	• Run the given command when the application stops. • <cmd line> is the full command line for the program to run, including any arguments. Please use quotes around each element if the command line contains spaces.
-of <file>	• Log the application's output (from stdout and stderr) to the given file. • <file> is the full path to the file where the output will be saved. Please use quotes if the name contains spaces. • This will work for console-based applications, not regular Windows applications.
-om <MB>	• When logging output, don't let the file size exceed the given <MB> value (in Mega-Bytes). • If the file grows to <MB> MB in size, the oldest ¼ will be discarded to make room for new data.
-j <description>	• Set the given description for the service. Be sure to enclose the text in quotes and limit the length to 200 characters.
-ps	• Prevent the computer from automatically going to sleep while the service is running.

Copyright © 2001-2015, Core Technologies Consulting, LLC

4.2.2. Logon

-l <account>	• <account> is the name of a user, under which the service shall run. • Note: This account needs to have the "Log on as a service", which can be granted from the "Control Panel / Administrative Tools / Local Security Settings" application (Local Security Policy on Windows 2000) . If not, it will not be able to start. • It is recommended that this user be a member of the “Administrators” group or the application may not be able to show its windows. • <account> should have the following format: "<Domain>\<user>", "Zone1\user4" for instance. • The domain "." is predefined as the local machine. So one might use ".\user1" too. • If not specified, the service/application will be run under the built-in “LocalSystem” account.
-p <password>	• The password for the user account under which the service shall run.
-lh	• Specify this flag to hide the application’s windows and task bar icons when a user logs on.

Copyright © 2001-2015, Core Technologies Consulting, LLC

4.2.3. Monitoring

-k	• Keep the program alive, restarting it when it crashes or exits unexpectedly. • AlwaysUp will log a message to the Windows Event Log whenever it restarts the application.
-c [<ms>]	• Check interval. <ms> is the number of milliseconds to pause in between checks to see if the program has crashed, or is displaying a message box. • Defaults to 2000 if not specified.
-m	• Dismiss common “application crashed” message boxes, logging their content to the event Log.
-ms	• When dismissing common “application crashed” message boxes, immediately stop the application when the most serious are encountered.
-z <mb>	• Stop the application when it grows larger <mb> megabytes.
-q <min>	• Stop the application if it “hogs” a CPU for longer than <min> minutes. • The application is designated a “CPU hog” if it uses more than 95% of the CPU over the given duration.
-v <min>	• Stop the application if it “hangs” for longer than <min> minutes.
-rm	• Stop the application when the computer resumes from a sleep/hibernation/low-power consumption state.
-y <day> <time> <reboot>	• Stop the application regularly, at a given time. • <day> is a numeric representation of the day: Monday = 1 Tuesday = 2 Wednesday = 3 Thursday = 4 Friday = 5 Saturday = 6 Sunday = 7 Daily = 8 (every day) • <time> represents a 24-hour time value in the format HH:MM:SS. Please be sure to specify the three numbers. For example, 3:30 AM is represented as “03:30:00” and 4 PM is “16:00:00”. • <reboot> indicates if the computer should be rebooted or not, with:

Copyright © 2001-2015, Core Technologies Consulting, LLC

	No = 0 Yes = 1
-ym <mb>	• When stopping the application at a given time, only stop it if it has grown larger than <mb> megabytes.
-ym <mb>	• When stopping the application at a given time, only stop it if it has grown larger than <mb> megabytes.
-yi <min>	• When stopping the application at a given time, only stop it if it has been idle for at least <min> minutes.
-yl	• When stopping the application at a given time, only stop it if no one is logged on.
-yt <min>	• When stopping the application at a given time, keep trying to stop for <min> minutes if not all conditions are met at the chosen time.
-da	• When the application stops, restart it on the next hour.
-db	• When the application stops, restart it on the next hour or half-hour.
-dc <seconds>	• When the application stops, restart it after the given number of <seconds>.
-dt <time>	• When the application stops, restart it at a specific time. • <time> must be in the 24-hour format "HH:MM:SS". For example “17:30:00" represents 5:30 PM.
-df	• Perform the delay on the first run, when AlwaysUp starts up and runs your application for the first time.
-ds	• Suspend the PC while waiting to start the application.
-dm	• Minimize writing to the event log (for applications that stop & restart frequently).
-ac <cmd> <i>	• Run the given automation script/program regularly. • <cmd> is the full command line of the script/program. Be sure to use quotes around each element if the command line contains spaces. See section 11.1 for special command-line values. • <i> specifies the interval (in seconds) between each run of the automation program. • See this page for sample AutoIt scripts: http://www.coretechnologies.com/products/AlwaysUp/AutoIt/
-an <num-times>	• Only run the automation program a few times after the application starts. • <num-times> is the number of times to run, after which the automation script will retire.
-fd <n> <m>	• Stop restarting the application if it fails <n> times in <m> minutes.

Copyright © 2001-2015, Core Technologies Consulting, LLC

4.2.4. Customized “Sanity Checks”

See Appendix I for details on the “Sanity Check” feature.

-t <cmd line>	• Perform a periodic "Sanity check" on the program being managed. The <cmd line> is the full command line of the sanity check program, e.g. "C:\apps\testServer.exe 10.10.10.10 4552". Please use quotes around each element if the command line contains spaces. • The program must complete with an error status of 0 to be considered successful. If unsuccessful, the application will be restarted. • See section 11.1 for special command-line values.
-a <seconds>	• Check sanity interval. <seconds>is the number of seconds to pause in between runs of the sanity check program. • Defaults to 30 if not specified.
-x <seconds>	• Check sanity timeout interval. Wait up to <seconds> seconds for a response from the check sanity program. If none is received, consider the result a failure. This will cause the program to be terminated. • The default value is 120 if not specified.
-xd <min>	• Delay for <min> minutes before running the first sanity check after the service starts. This may be useful if it takes a while for your service/application to be ready
-xr	• Apply the sanity check delay on each restart, not just on the first run. Only valid with the -xd flag.

Copyright © 2001-2015, Core Technologies Consulting, LLC

4.2.5. Notification (via E-Mail)

-o <address>	• <address> is the the “To” address for all email messages. Multiple addresses can be specified by separating them with a comma and placing the whole value in quotes, for example: “admin@myserver.com, system@myserver.com”. • (Note: this is the letter ‘o’, not a zero)
-g <smtphost>	• <smtphost> is the name or IP address your SMTP mail server. For example “mail.mydomain.com” or “smtp.gmail.com”. • Defaults to “mail” if not specified.
-h <address>	• <address> will be placed in the "From:" field of mail sent via SMTP. For example, brian@mydomain.com • If not specified, the mail sent may not have a "From:" address.
-3 <name>	• The real name to be placed in the "From:" field of email sent via SMTP. For example, “Brian Smith”. • If not specified, the email sent may not have a "From:" name.
-b	• Send an email message when an “application crashed” message box has been dismissed.
-r	• Send an email message when the application crashes or stops prematurely and has to be restarted.
-rs	• Send an email message when the service starts.
-rb	• If sending an email when the service starts (with the -rs option) only send an email if the startup is because the machine has booted • Only meaningful with the -rs flag.
-4 <port>	• The port number associated with the SMTP mail server. • Defaults to 25 if not specified.
-5	• Indicates that the SMTP server uses SSL to send email.
-5s	• Indicates that the SMTP server uses STARTTLS to send email.
-6	• Indicates that the SMTP server requires authentication. • An authentication method must be specified (see below)
-7 <auth>	• Specifies the authentication method to be used when sending email. • <auth> is a numeric representation of the authentication method:

Copyright © 2001-2015, Core Technologies Consulting, LLC

	CRAM-MD5 = 1 AUTH-LOGIN = 2 AUTH-PLAIN = 3 NTLM = 4 • All authentication methods except NTLM require a login and password (see below)
-8 <login>	• Specifies the user name/login to access the SMTP mail server.
-9 <password>	• Specifies the password to access the SMTP mail server.
-tm	• Send emails in plain text format instead of HTML.
-et <text>	• Additional text to be included in each email message sent. Please specify at most 30 characters and please put quotes around the value.

Copyright © 2001-2015, Core Technologies Consulting, LLC

4.2.6. Startup

-rn	• Automatically restore any network drives accessible to the user who will run the application. • Only works for drives mapped automatically at logon, without the need for the user to type in a password. • May take some time at startup if drive access is “slow”.
-nt	• Ensure that the Windows networking components/services are up and running prior to starting the application.
-pt <s>	• Wait for <s> seconds before starting the application.
-pr	• Only wait (with the –pt option) only when the service has been started shortly after a reboot.
-1 <cmd line>	• Run the given program at startup, prior to running the application. • <cmd line> is the full command line for the program to run, including any arguments. Please use quotes around each element if the command line contains spaces.
-11	• In addition to at startup, run the startup program (given with the -1 option above) whenever the application is restarted.
-2 <deps>	• Ensure that the given services are started prior to starting the application. • <deps> is the list of service names, available from the Control Panel Services application. Separate multiple names with a comma (,), for example “LmHosts,Netman”
-is	• Before first starting the application/service, stop all instances of the configured application.
-ir	• Stop all instances of the configured application before restarting it. • Only relevant when the -is option is specified.

4.3. Return Codes

InstallService returns 0 (zero) on success and non-zero on failure. When it fails, the reason for failure is reported to the standard output (stdout).

Copyright © 2001-2015, Core Technologies Consulting, LLC

4.4. Examples

1. The following installs the "Notepad" application as a service, which runs notepad.exe on the specified text file, restarting it whenever it crashes or stops unexpectedly:

   InstallService Notepad "c:\winnt\notepad.exe c:\data.txt" -k

2. The following installs the “MyServer” service, which runs “C:\MyServer\MyServer.exe /fast /usermode true” from directory "C:\MyServer\Run" as user “tbrown” with password "thepassword". The application will be restarted whenever it crashes or stops unexpectedly, whenever it uses more than 512MB of RAM, and every day at 3AM. The application will be started only after the “Netman” (Network Connections) service has started. E-mail will be sent to admin@mydomain.com using server "mail.mydomain.com" whenever the application is restarted, or whenever a message box is dismissed:

   InstallService.exe "My Server"
   "C:\MyServer\MyServer.exe /fast /usermode true"
   -k -m -o admin@mydomain.com
   -h alwaysup_report@mydomain.com
   -g "mail.mydomain.com" -r -b -l ".\tbrown"
   -p "thepassword"
   -w "C:\MyServer\Run" -y 8 3 -z 512 -2 "Netman"

4.5. Using the AlwaysUp GUI to Create the Command Line

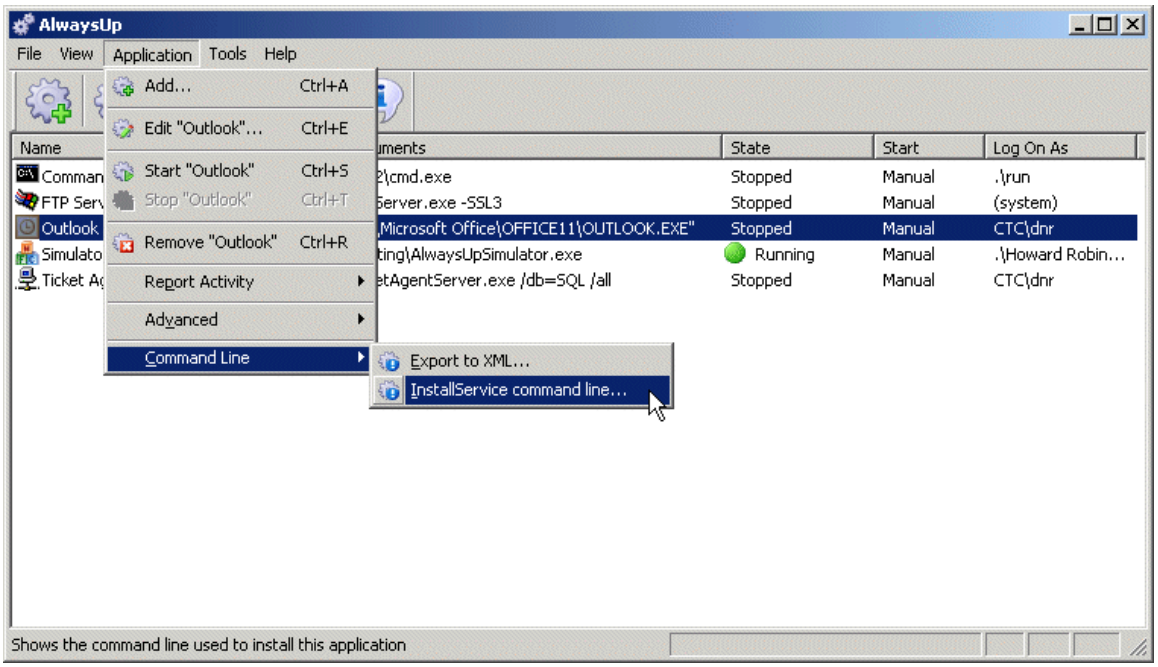
The AlwaysUp GUI can be used to ease the creation of the InstallService command line. Basically, you configure your application/service in the AlwaysUp GUI and select a special OEM menu item to reveal the command line used to install the service.

To use this method:

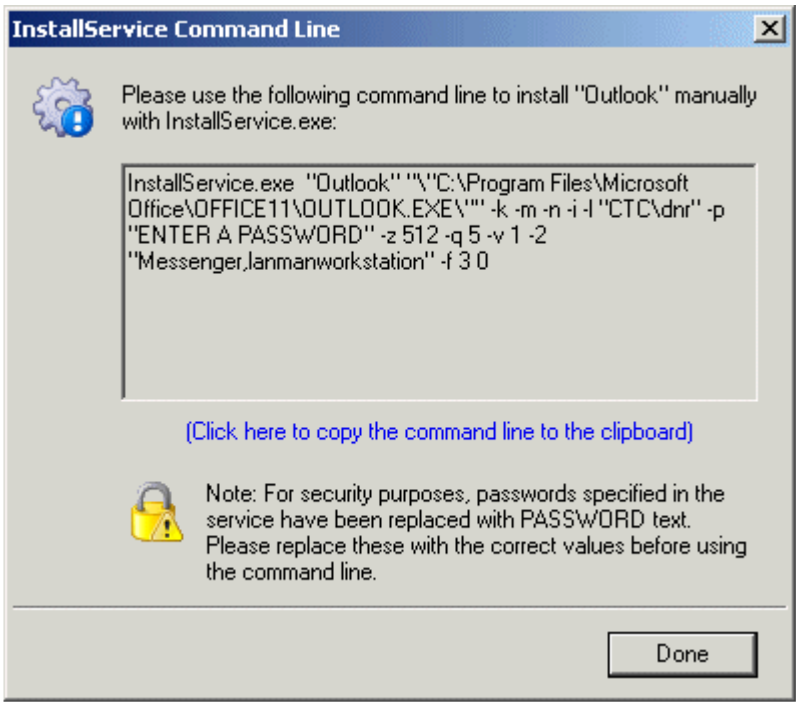
1. Install the AlwaysUp GUI package (available at http://www.coretechnologies.com/products/AlwaysUp/)
2. Copy your license file (“alwaysup.dat”) to the location where AlwaysUp was installed (by default, “C:\Program Files\AlwaysUp”).
3. Start AlwaysUp
4. Create your service/application

Copyright © 2001-2015, Core Technologies Consulting, LLC

5. Highlight the service in the list and select “Command Line | InstallService command line...” from the “Application” menu:



6. The following window will be presented:


InstallService Command Line
Please use the following command line to install "Outlook" manually with InstallService.exe:
InstallService.exe "Outlook" "\"C:\Program Files\Microsoft Office\OFFICE11\OUTLOOK.EXE\"" -k -m -n -i -l "CTC\dnr" -p "ENTER A PASSWORD" -z 512 -q 5 -v 1 -2 "Messenger,lanmanworkstation" -f 3 0
(Click here to copy the command line to the clipboard)
Note: For security purposes, passwords specified in the service have been replaced with PASSWORD text. Please replace these with the correct values before using the command line.
Done

Copyright © 2001-2015, Core Technologies Consulting, LLC

7. Copy the command line from the window by clicking on the hyperlink text.

Note that the “Command Line” menu is NOT available on machines without a command-line license (as specified in the alwaysup.dat license file).

4.6. Using the AlwaysUp GUI to Create an XML file

The AlwaysUp GUI can export an existing AlwaysUp service to an XML file, which can later be used to install the service on another machine.

To use this method:

1. Install the AlwaysUp GUI package (available at http://www.coretechnologies.com/products/AlwaysUp/)
2. Copy your license file (“alwaysup.dat”) to the location where AlwaysUp was installed (by default, “C:\Program Files\AlwaysUp”).
3. Start AlwaysUp
4. Create your service/application
5. Highlight the service in the list and select “Export…” from the “Application” menu.
6. This will bring up a standard “Save As” dialog box where you specify the name of the XML file to be written.

In addition, the AlwaysUp GUI supports simple drag-and-drop from the list of applications to Windows Explorer (or the desktop, etc.) to create a suitable XML file.

Copyright © 2001-2015, Core Technologies Consulting, LLC

5. UninstallService

UninstallService will remove a service previously installed with InstallService.

5.1. Command Line Syntax

The command line syntax for UninstallService is:

UninstallService.exe <Service name>

Where:

<Service name> is the name of an existing service that was installed using InstallService.

5.2. Return Codes

UninstallService returns 0 (zero) on success and non-zero on failure. When it fails, the reason for failure is reported to the standard output (stdout). Note that the service will be stopped prior to uninstalling it, so it is not necessary to explicitly stop it before uninstalling.

5.3. Examples

The following would uninstall the "Notepad" service created in section 2.3:

UninstallService Notepad

5.4. Using SC

The “SC” command can also be used to uninstall the service. Please see the following pages for details on SC:

http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/sc.mspx?mfr=true

http://www.ss64.com/nt/sc.html

Copyright © 2001-2015, Core Technologies Consulting, LLC

When using SC, you must specify the service name with a special suffix added by AlwaysUp when it was installed. For example, if your service is called “Notepad”, the name of the service should be “Notepad (managed by AlwaysUpService)”. You would invoke SC as:

SC.exe delete “Notepad (managed by AlwaysUpService)”

Copyright © 2001-2015, Core Technologies Consulting, LLC

6. AlwaysUpService

AlwaysUpService is an internal component distributed with the suite. It is responsible for actually running your application as a service and can be invoked to export an application's configuration as XML, or to start your application in the current session

6.1. Export to XML

The command line syntax to export an application's configuration as XML is:

AlwaysUpService.exe <Service name> -exportxml <XML file name>

Where:

<Service name> is the name of an existing service that was installed using InstallService.

<XML file name> is the full path to a file name in which to save the XML. The file is overwritten if it already exists.

Please see Appendix II for details on the XML.

6.2. Start or Restart your Application in the Current Session

If you wish to start or restart your application in the session that you are currently logged in to, the syntax is as follows:

AlwaysUpService.exe <Service name> -start-in-current-session

or

AlwaysUpService.exe <Service name> -restart-in-current-session

Where:

<Service name> is the name of an existing service that was installed using InstallService.

Copyright © 2001-2015, Core Technologies Consulting, LLC

Please see Appendix III for a discussion of Sessions and why you may want to start your application in this manner.

Copyright © 2001-2015, Core Technologies Consulting, LLC

7. Starting and Stopping AlwaysUp Services with the NET Command

AlwaysUp Service can be manipulated using the “net.exe” command (just as any other service can be).

7.1. To Start a Service

Run:

net start "<ServiceName> (managed by AlwaysUpService)”

Where:

<ServiceName> is the name give to the service at installation

Notice that you must add the “(managed by AlwaysUpService)” suffix as that is how the service is actually named in the Service Control Manager.

For example, to start an AlwaysUp service named “Server”, use:

net start "Server (managed by AlwaysUpService)”

The application will be started in the default session, and will be suitable for most applications (and all of those without a user interface).

7.2. To Start a Service in a Specific Session

On systems with multiple sessions, AlwaysUp can be directed to launch the application in a specific session, making the applications windows available to the user logged in there.

Run:

net start "<ServiceName> (managed by AlwaysUpService)” /<SID>

Where:

<ServiceName> is the name give to the service at installation

<SID> is the ID of the session where the application is to be run

Copyright © 2001-2015, Core Technologies Consulting, LLC

Notice that you must add the “(managed by AlwaysUpService)” suffix as that is how the service is actually named in the Service Control Manager.

For example, to start an AlwaysUp service named “Server” in session 2, use:

net start "Server (managed by AlwaysUpService)" /2

Please see Appendix III for a discussion of Sessions and why you may want to start your application in this manner.

7.3. To Start a Service in a Specific User’s Session

On systems with multiple sessions, AlwaysUp can be directed to launch the application in a user’s session, making the applications windows available to him.

Run:

net start "<ServiceName> (managed by AlwaysUpService)" /"user:<User-Name>"

Where:

<ServiceName> is the name give to the service at installation

<User-Name> is the user name/login of the user logged in to the session where the application is to be run

Notice that you must add the “(managed by AlwaysUpService)” suffix as that is how the service is actually named in the Service Control Manager.

For example, to start an AlwaysUp service named “Server” in user mjones’s session (so that the application is visible to him), use:

net start "Server (managed by AlwaysUpService)" /"user:mjones"

Please see Appendix III for a discussion of Sessions and why you may want to start your application in this manner.

7.4. To Stop a Service

Copyright © 2001-2015, Core Technologies Consulting, LLC

Run:

net stop "<ServiceName> (managed by AlwaysUpService)"

Where:

<ServiceName> is the name give to the service at installation

Notice that you must add the "(managed by AlwaysUpService)" suffix as that is how the service is actually named in the Service Control Manager.

For example, to start an AlwaysUp service named "Server", use:

net stop "Server (managed by AlwaysUpService)"

Additional details on the "net" command are available at:

http://www.ss64.com/nt/net_service.html and
http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/net_subcmds.mspx

Copyright © 2001-2015, Core Technologies Consulting, LLC

8. Files to Distribute with your Applications

These files should be distributed with your applications:

File name	Description
AlwaysUpService.exe	The main executable for running your application.
InstallService.exe	Used to install the service. It can be deleted after completing your installation process.
UninstallService.exe	Used to uninstall the service. Only necessary if you provide a way to uninstall your application.
EventMessageFile.dll	Supporting DLL
GHCrUtils.dll	Supporting DLL

Copyright © 2001-2015, Core Technologies Consulting, LLC

9. Integrating AlwaysUp CLT into your Application's Installer

AlwaysUp CLT was designed to be easily deployed with today's most popular installation software (Windows Installer, InstallShield, Inno Setup, etc.). While each Installer has its own way of doing things, the following steps must be performed to distribute AlwaysUp seamlessly alongside your application:

1. Copy the AlwaysUp executables and supporting DLLs to a directory of your choice. (Section 8 lists the files.) If you have already purchased AlwaysUp CLT, please include your .DAT license file as well.

2. Run "InstallService.exe" (described in Section 4) with the parameters necessary to configure AlwaysUp CLT to run your application as a service.

3. Start your application, if desired (see Section 7).

That's it! After installation, AlwaysUp CLT will run your application as configured in step 2.

Please don't hesitate to get in touch with us if you need help integrating AlwaysUp with your application's installer. We will be happy to assist.

Copyright © 2001-2015, Core Technologies Consulting, LLC

10. Licensing & Registration

The AlwaysUp command-line components are free to evaluate for the first 30 days. After the trial period a license must be purchased to continue usage.

If you find the programs useful, we encourage you to license them. The small licensing fee charged will fund continued development of AlwaysUp and will entitle you to expedited support from Core Technologies Consulting LLC, the author of the software.

Please send email to support@CoreTechnologies.com for licensing details.

Copyright © 2001-2015, Core Technologies Consulting, LLC

11. Troubleshooting and Reporting Problems

Please consult the AlwaysUp FAQ for troubleshooting tips and answers to frequently asked questions:

http://www.coretechnologies.com/products/AlwaysUp/AlwaysUp_FAQ.html

If you encounter a problem while using AlwaysUp, please send email to:

support@CoreTechnologies.com

Be sure to include the following information:

- Your Operating System
- The version of AlwaysUp in use
- Detailed steps for reproducing any software bugs/issues

Feel free to send requests for enhancements to the same address.

Copyright © 2001-2015, Core Technologies Consulting, LLC

12. Appendix I: The Sanity Check Utility

AlwaysUp can periodically run a program (a “Sanity check utility”) to periodically check if the application being monitored is functioning normally. The utility should exit with a return code of:

- 0 when the check succeeds;
- 1 when the check fails and the application should be stopped and restarted as you have configured;
- 10 when the check fails and the computer should be rebooted;
- 100 when the check fails and your application should be stopped **and not restarted**;
- any other value when the check fails due to an error independent of the application being monitored (such as an error internal to the utility), or to simply indicate failure without causing a restart.

In the third case, the application is not restarted but a message is written to the event log (and an email is sent if so configured).

Note that if the sanity check utility fails to complete in 120 seconds, the application will be restarted.

The popular “ConnectToSocketSanityCheck” program (with full source code) is available in the “SanityCheck” sub-directory. Given a host and port on the command line, it returns 0 if a socket could be opened, 1 if the socket could not be opened, and -1 if there was an internal error initializing the sockets API. It can be used to check if a TCP/IP application, such as a web server, is accepting connections.

Copyright © 2001-2015, Core Technologies Consulting, LLC

12.1. Special Command Line Variables

AlwaysUp is able to pass your sanity check program (and any of the other commands invoked to support your application) one or more “special” values. Compose your command line with the appropriate string and AlwaysUp will make the substitution before invoking your program.

Replacement String	Replaced With
$ALWAYSUP_PID	The program identifier (PID) of your running application, as seen in the Task Manager, or -1 if the application is not running.
$ALWAYSUP_EXENAME	The name of the executable run to invoke your application. (Just the name, not the full path.)
$ALWAYSUP_APPNAME	The name of your application in AlwaysUp.
$ALWAYSUP_APPSTARTTIME	The time when the application being monitored was started, in the fixed format “YYYY/MM/DD HH:MM:SS". This value is the empty string if the application is not running.
$ALWAYSUP_APPUPTIME	The number of seconds that have elapsed since the application being monitored was started. This value is -1 if the application is not running.
$ALWAYSUP_SERVICENAME	The name of the service created by AlwaysUp. This will be the name of your application, plus a suffix of “ (managed by AlwaysUpService)”.
$ALWAYSUP_SERVICESTARTTIME	The time when the service created by AlwaysUp was started, in the fixed format “YYYY/MM/DD HH:MM:SS".
$ALWAYSUP_SERVICEUPTIME	The number of seconds that have elapsed since the service created by AlwaysUp was started.
$ALWAYSUP_RUNNUMBER	The number of times AlwaysUp has started your application. This will be 0 before AlwaysUp starts your application and will increase by 1 each time that AlwaysUp starts/restarts your program.

Copyright © 2001-2015, Core Technologies Consulting, LLC

For example, to have AlwaysUp pass your Sanity check program the application's program identifier (PID), then your command line might resemble this:

C:\myserver\my_check.exe $ALWAYSUP_PID

If your main application is running with PID 563, then your Sanity program will be invoked like this:

C:\myserver\my_check.exe 563

12.2. Checking a Web Server with http-ping

Our free http-ping utility can also be used to ensure that your web server is serving pages. This sanity check script will use http-ping to try to download a given page from your server and will signal AlwaysUp to restart your application if the download fails:
http://www.coretechnologies.com/products/AlwaysUp/check-web-server-sanity-check.bat
Please edit the script to specify the address of your own web server/page before using it!

Find out more about http-ping here:
http://www.coretechnologies.com/products/http-ping/

12.3. Other Sanity Check Utilities

We have created a few Sanity Check programs that can be easily downloaded and used free of charge. Please visit our web site to see what is available:
http://www.coretechnologies.com/products/AlwaysUp/Plugins/

If you don't find what you need there, please send us an email describing in detail what your Sanity Check should do and we will try our best to create one for you.

Copyright © 2001-2015, Core Technologies Consulting, LLC

13. Appendix II: The InstallService XML

As outlined in section 2.1, the InstallService.exe utility can accept XML describing how the application is to be installed. The XML is governed by the schema available at our web site: http://www.coretechnologies.com/products/AlwaysUp/AlwaysUpCLT/alwaysup_installservice.xsd

XML to install the Windows “Notepad” application with AlwaysUp follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<alwaysup-installservice-configuration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://www.coretechnologies.com/AlwaysUp/AlwaysUpCLT/a
lwaysup_installservice.xsd">

	<!-- The name of the service -->
	<name>Notepad Test</name>

	<!-- An optional description of the service of the service -->
	<description>Installed via XML using AlwaysUp</description>

	<!-- General settings -->
	<general>
		<!-- Path to Notepad executable. This may be different on your system! -->
		<application-path>c:\winnt\notepad.exe</application-path>
		<application-arguments></application-arguments>
		<start-directory></start-directory>
		<set-priority priority="normal"/>
		<start type="manually"/>
	</general>

	<!-- Log-on settings. -->
	<log-on-as-system/>

	<!-- Monitoring the application -->
	<monitor>
		<restart>
			<when-app-crashes/>
			<when-app-hangs enabled="true" for-longer-than="1"/>
			<when-memory-exceeds enabled="true" size="32"/>
			<when-cpu-hogged enabled="true" for-longer-than="5"/>
		</restart>
		<dismiss-messageboxes/>
	</monitor>

	<!-- Notification settings. Change to reflect your mail settings. -->
	<notification>
		<email>
```

Copyright © 2001-2015, Core Technologies Consulting, LLC

```
                        <to-address>admin@yourdomain.com</to-address>
                        <from-name>Administrator</from-name>
                        <from-address>alwaysup@yourdomain.com</from-address>
                        <send>
                                <when-application-stops/>
                                <when-messagebox-dismissed/>
                                <daily/>
                                <weekly/>
                        </send>
                        <smtp-server>
                                <name>mail.yourdomain.com</name>
                                <port>25</port>
                        </smtp-server>
                </email>
        </notification>

        <!-- Startup settings -->
        <startup>
                <dependencies enabled="true">
                        <service-name>lanmanworkstation</service-name>
                </dependencies>
        </startup>

</alwaysup-installservice-configuration>
```

This file is also available at:
http://www.coretechnologies.com/products/AlwaysUp/AlwaysUpCLT/install_notepad_alwaysup_sample.xml

Copyright © 2001-2015, Core Technologies Consulting, LLC

14. Appendix III: Remote Desktop and Terminal Services Sessions

Whenever a user logs in to a Windows PC, he interacts with a "session" – his own private graphical sandbox with a dedicated desktop and set of windows. PCs with Terminal Services and Remote Desktop software allow multiple users to be logged in to a single PC simultaneously, with each user having his own private session and the illusion of being the only user of the PC. This privacy ensures that application windows in one session are not visible in (nor accessible to) another session. Each session is identified by a unique session ID.

Windows services can be affected by PCs supporting multiple sessions. When the PC boots and services are started, only one session is created and all services will run in that session. Called the "console session", or "session 0", this special session may be handled differently by each version of Windows. For example, on Windows 2000, XP and 2003 a user can log in to the console session, but on Windows Vista or Server 2008 it is isolated for security purposes (see "*Impact of Session 0…*" below for details).

When AlwaysUp is running your application in the console session it is not visible in any other session. If you start a Remote Desktop Connection and you are granted another session (with a different ID), you will not be able to interact with your application running in the console session. Tricks such as starting Remote Desktop with the "/console" flag will allow you to connect to the console session on Windows XP and 2003, but will not do the job on Windows Vista or Server 2008 as the console session is isolated there.

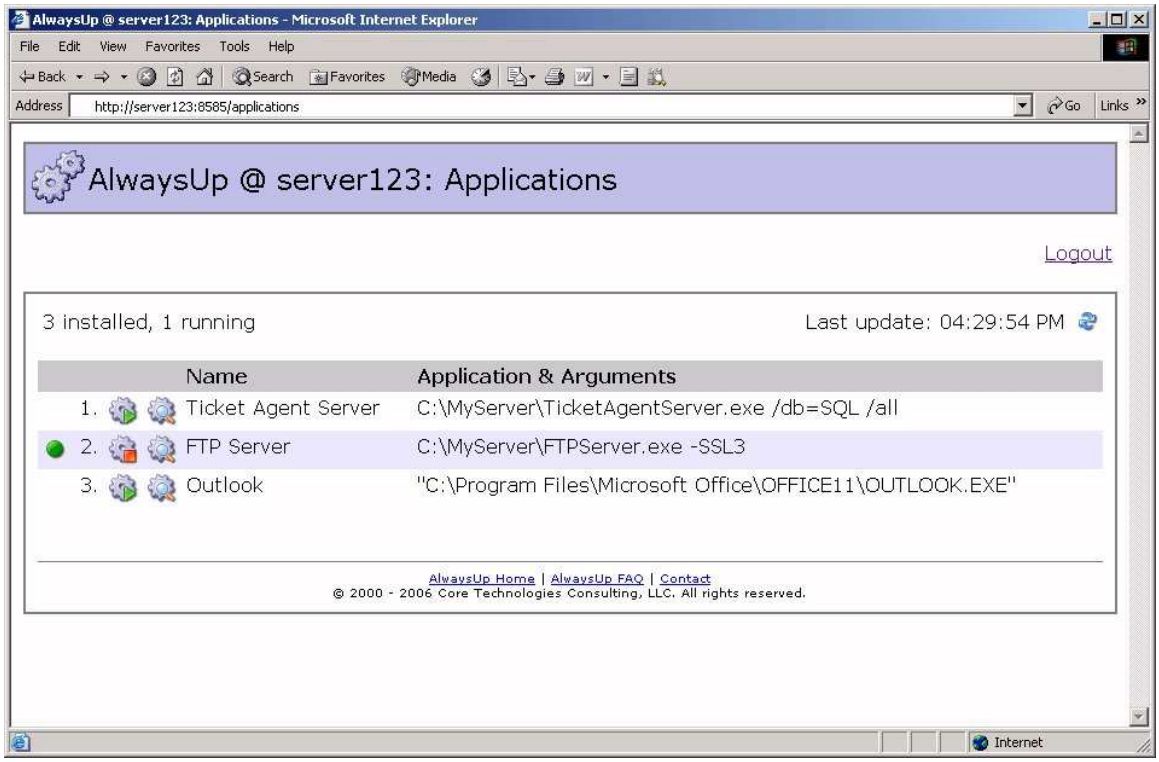
If your application is running in a different session, you can use AlwaysUp to run it in the current session and make its windows and tray icons available to you by selecting "Start <app> in this session" or "Restart <app> in this session" from the "Application" menu. Note that restarting will temporarily stop your application.

Additional information:
What "sessions" are on Windows NT:
http://homepages.tesco.net/~J.deBoynePollard/FGA/windows-nt-session.html
Terminal Services Sessions:
http://msdn2.microsoft.com/en-us/library/aa383496.aspx
How Terminal Services Works:
http://technet2.microsoft.com/windowsserver/en/library/2cb5c8c9-cadc-44a9-bf39-856127f4c8271033.mspx?mfr=true
Impact of Session 0 Isolation on Services and Drivers in Windows Vista
http://www.microsoft.com/whdc/system/vista/services.mspx

Copyright © 2001-2015, Core Technologies Consulting, LLC

15. Appendix IV: AlwaysUp Web Service

AlwaysUp Web Service is a free add-on program that allows you to control AlwaysUp from your web browser. Find out more and download the latest version at:
http://www.CoreTechnologies.com/AlwaysUp/AlwaysUpWebService



Copyright © 2001-2015, Core Technologies Consulting, LLC

16. Appendix V: AlwaysUp Event Log Messages

As is the convention with Windows Services, AlwaysUp reports information, warnings, and errors to the Windows Application Event Log. That log is accessible through the Control Panel Event Viewer applet.

Each entry is logged on behalf of the related AlwaysUp application. For example, if you have an application called **"MyServer"**, then the Event Log will have entries with **"MyServer (managed by AlwaysUpService)"** as the designated source.

The following page presents the various messages that can be logged, along with more detailed information for each situation:

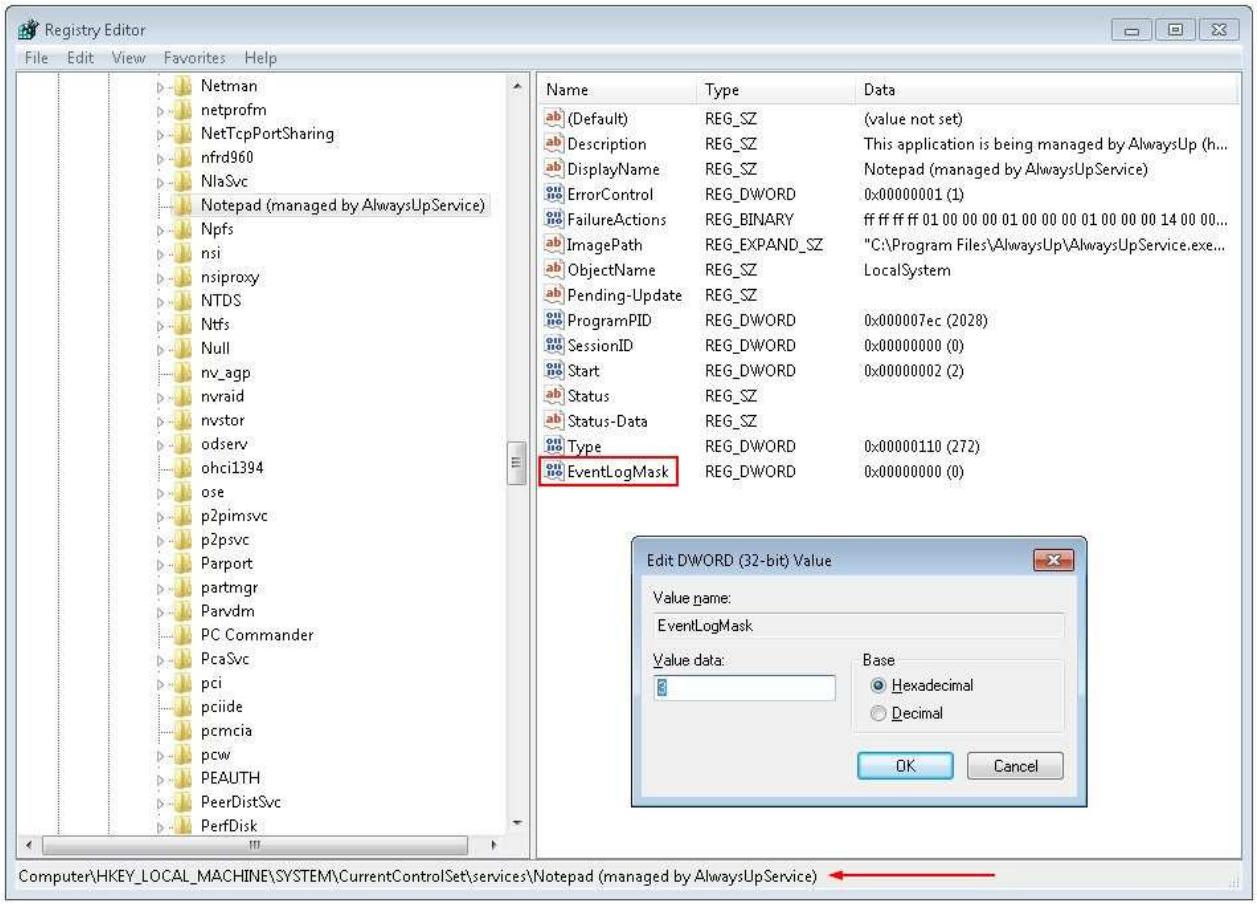
http://www.coretechnologies.com/products/AlwaysUp/AlwaysUp_EventLog.html

16.1. How to Suppress Writing to the Event Log

Event logging can be entirely disabled or limited by category via a registry key. To do so for a specific application/service:

1. Start the Registry Editor, Regedit.exe.
2. Navigate to **HKEY_LOCAL_MACHINE\ SYSTEM\CurrentControlSet\services**
3. In the tree on the left, click on the service that you wish to modify. If your application is called “Notepad”, then look for “Notepad (managed by AlwaysUpService)”.
4. From the menu, select “Edit > New > DWORD (32-bit) value”.
5. Rename the new value on the right to **“EventLogMask”.**
6. Right-click on that name and select “Modify...”
7. Set the value to:
   - **0** to disable all writing to the event log
   - **1** if you want only errors
   - **3** if you want errors and warnings
   - **7** if you want errors, warnings and information (the default)
8. Click OK to record the value.
9. Close the Registry Editor.

Copyright © 2001-2015, Core Technologies Consulting, LLC



The changes will take effect the next time you restart your application in AlwaysUp (or reboot your PC).

Note that you can affect all AlwaysUp applications by setting the EventLogMask value in **HKEY_LOCAL_MACHINE\SOFTWARE\ Core Technologies Consulting\AlwaysUp** (or **HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Core Technologies Consulting\AlwaysUp** on 64-bit machines) instead. However an individual service's setting, when present, will override that master setting.

Copyright © 2001-2015, Core Technologies Consulting, LLC